Mixed-integer Programming based Differential and Linear Cryptanalysis

Siwei Sun

State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences

Data Assurance and Communication Security Research Center,

Chinese Academy of Sciences, Beijing, China

A joint work with

Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi,

Ling Song, Kai Fu

ASK 2015 @ Singapore

Outline

- Introduction
- Mixed-Integer Programming (MIP) based Automatic Differential (Linear) Characteristic Search
- Applications, Improvements, and Adaptation for Specific Ciphers
- Future work: *Domain Specific Language (DSL)* for Constraint Programming based Automatic Cryptanalysis

Introduction

- Differential cryptanalysis [1] is one of the most powerful attacks on block ciphers
  - ✓ Truncated differential attack
  - ✓ Related-key differential attack
  - ✓ Boomerang attack
  - ✓ …

- Finding a good (related-key) differential (characteristic) with high probability is the first step in the (related-key) differential attack

[1] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. Journal of Cryptology, 4(1):3-72, 1991.

Existing methods for characteristic search

- Matsui's algorithm and its variants
  - ✓ Mitsuru Matsui, *On correlation between the order of S-boxes and the strength of DES*, Eurocrypt 1994.
  - ✓ Alex Biryukov, Ivica Nikolic.: *Search for related-key differential characteristics in DES-like ciphers*. FSE 2011
  - ✓ … other branch and bound based algorithms

- SMT(Satisfiability Modulo Theory) based Methods
  - ✓ Nicky Mouha and Bart Preneel. *Towards finding optimal differential characteristics for ARX: Application to Salsa20*. IACR Cryptology ePrint Archive, Report 2013/328, 2013.
  - ✓ Jean-Philippe Aumasson, Philipp Jovanovic, Samuel Neves. *Analysis of NORX: Investigating Differential and Rotational Properties*. In Latincrypt 2014, 2014.
  - ✓ Stefan Kölbl and Gregor Leander and Tyge Tiessen. *Observations on the SIMON block cipher family*. CRYPTO 2015.

- Integer programming based method

Existing methods for characteristic search

Integer programming based methods

- ✓ Nicky Mouha. *Differential and linear cryptanalysis using mixed-integer linear programming*. Information Security and Cryptology. Springer Berlin Heidelberg, 2012.
- ✓ Shengbao Wu, Mingsheng Wang. *Security Evaluation against Differential Cryptanalysis for Block Cipher* IACR ePrint 2011/551.
- ✓ Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Ling Song. *Automatic Security Evaluation and (Related-key) Differential Characteristic Search : Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-oriented Block Ciphers.* Asiacrypt 2014.
- ✓ Siwei Sun, Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi, Ling Song: *Towards Finding the Best haracteristics of Some Bit-oriented Block Ciphers and Automatic Enumeration of (Related-key) Differential and Linear Characteristics with Predefined Properties.* IACR Cryptology ePrint Archive 2014:747 (2014)

Mixed-integer programming (MIP), an example

- ✓ Objective function
- ✓ Constraints

$$
\begin{aligned}
&\min -x_1 + x_2 - 2x_3 + x_4 - x_5 \\
&\text{subject to} \\
&\qquad \begin{aligned}
x_1 + x_2 &\leq 1 \\
x_1 - 5x_2 + x_3 &\leq 2 \\
2x_3 + 2x_4 - 4x_5 &\leq 1 \\
x_2 - 2x_4 + x_5 &\leq 0 \\
x &\in \{0,1\}^5
\end{aligned}
\end{aligned}
$$

Feasible region: the set of all solutions satisfying the constraints

Mixed-integer programming based method

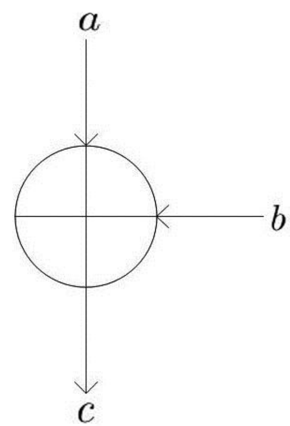
□ Basic Idea

- ✓ Describe the propagation characteristic of the difference patterns using linear inequalities

- ✓ Find solutions of the MIP model corresponding to differential characteristics with specific properties
  - ➢ High probability
  - ➢ Fixed input/output difference
  - ➢ Predefined number of active S-boxes
  - ➢ Predefined Hamming weight of the input/output differences
  - ➢ …

Constraints imposed on XOR

- Constraints imposed on the input and output differences by XOR



$$a=0, b=0 \rightarrow c=0$$
$$a=0, b=1 \rightarrow c=1$$
$$a=1, b=0 \rightarrow c=1$$
$$a=1, b=1 \rightarrow c=0$$

- Constraints (where d is a dummy variable and all variables are 0-1)

$$\begin{cases} a+b+c \geq 2d \\ d \geq a \\ d \geq b \\ d \geq c \\ a+b+c \leq 2 \end{cases}$$

eliminate the case of one and only one of a, b, and c is 1

eliminate the case of a=1, b=1 and c=1

- Constraints imposed on the input and output differences by an $m \times n$ S-box (<u>not necessarily invertible</u>)
  - ✓Let $x_1, x_2, \ldots, x_m$ be the input difference, and $y_1, y_2, \ldots, y_n$ be the output difference
  - ✓Let A be the variable indicating the activity of the S-box

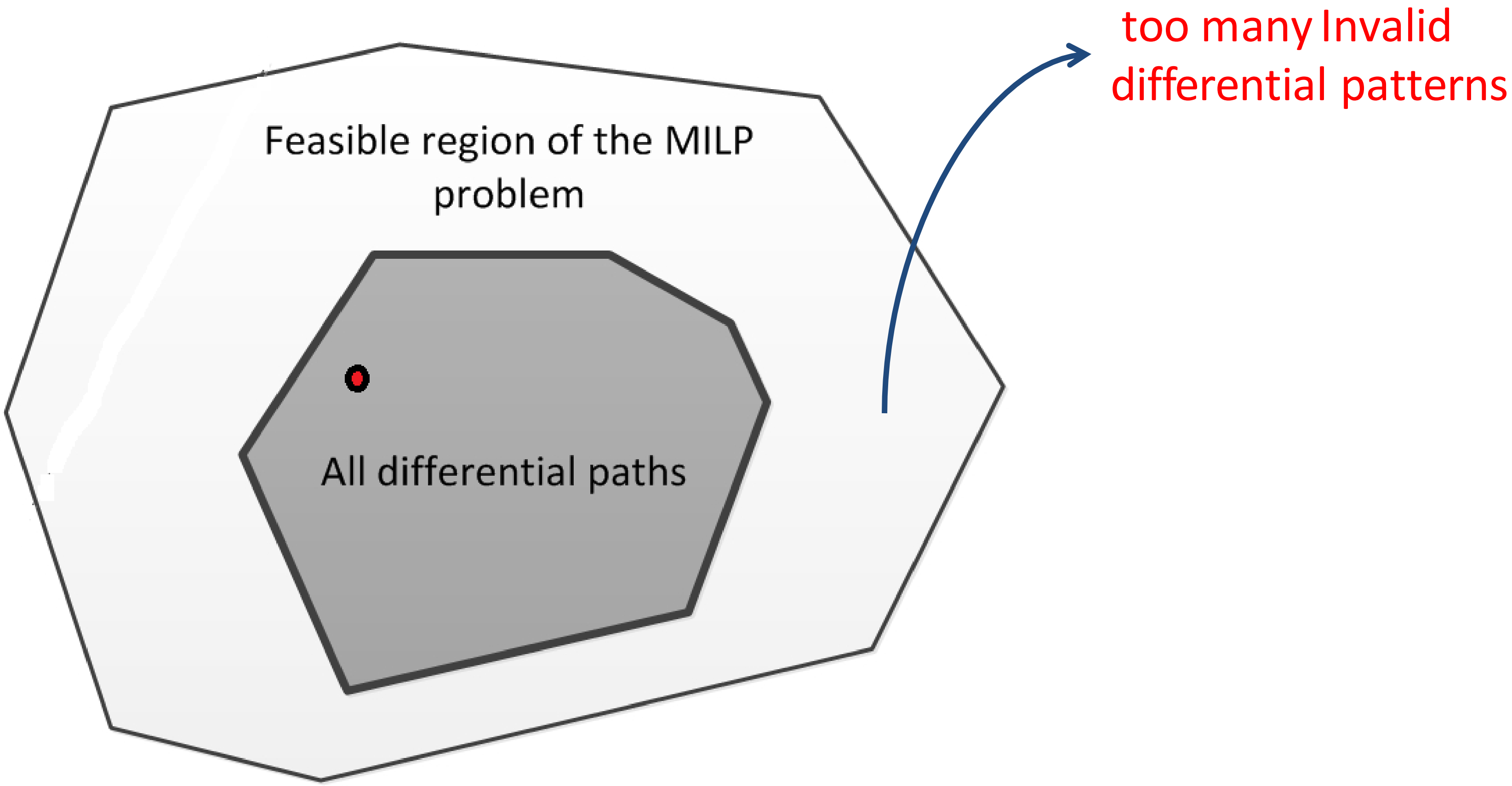
$$x_1 + \ldots + x_m - A \geq 0$$

At least one of the input difference bit x_i must be 1 if A = 1.

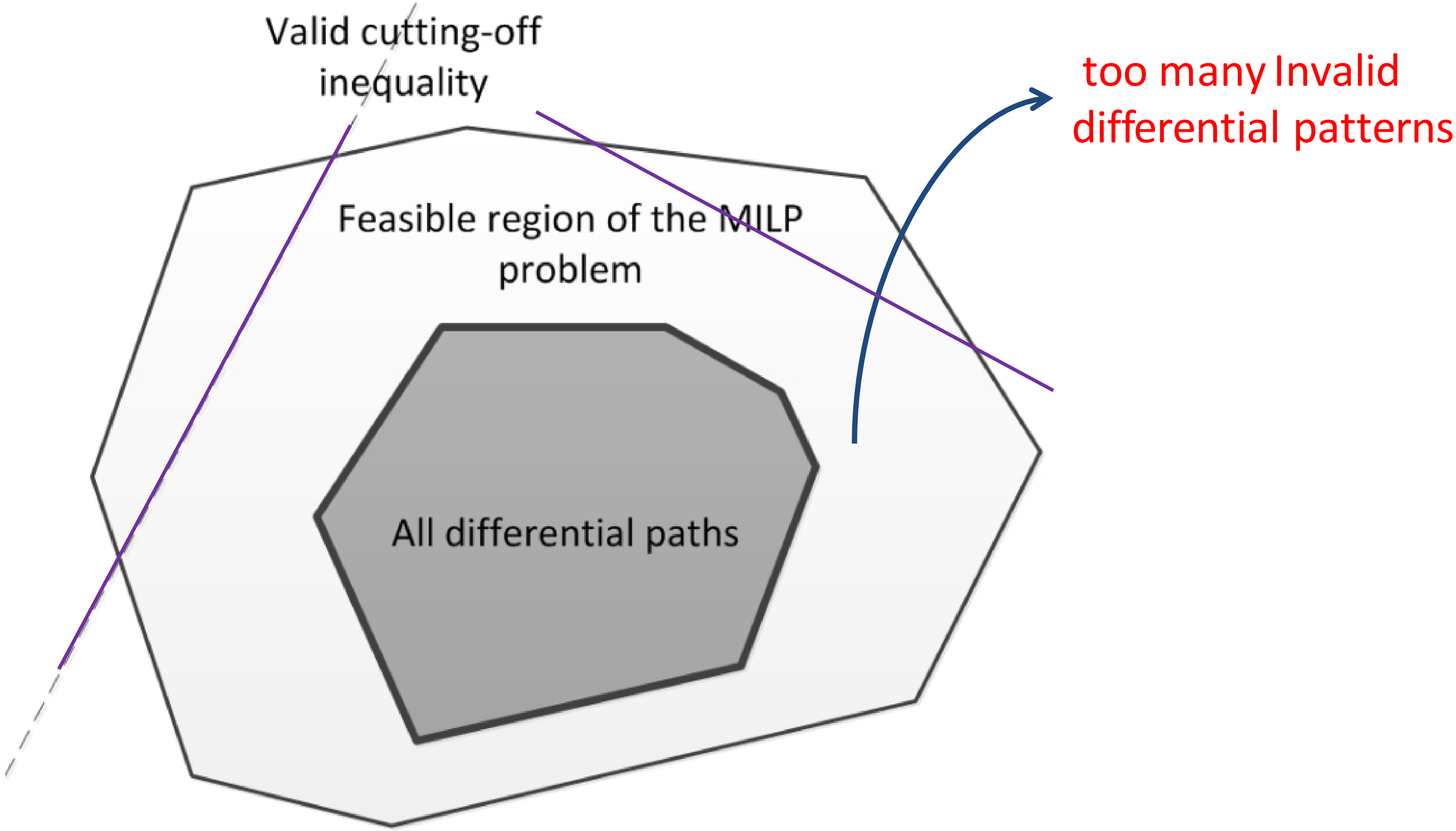
$$\begin{cases} A - x_1 \geq 0 \\ \ldots \\ A - x_m \geq 0 \end{cases}$$

A must be 1 (active), when anyone of the input difference x_i is 1.

- However, this is too coarse to describe an specific S-box, and result in an feasible region contain many invalid differential patterns



- Hence, we need the so called valid cutting-off inequalities to remove some impossible differential patterns of an specific S-box.

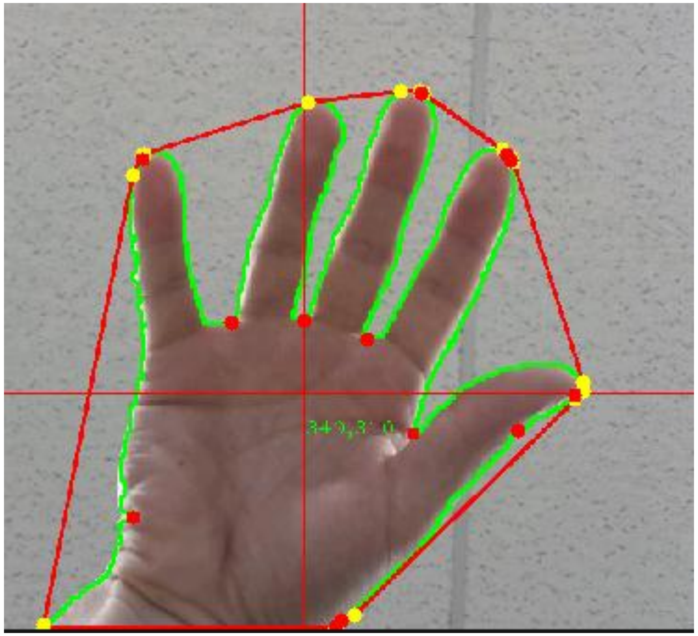


- Method for generating valid cutting-off inequalities for an specific S-box

Convex hull computation

☐ Convex hull computation

✓ Convex hull of a set of points in R^n : the smallest convex set that contains these points.



- Convex hull computation
  - ✓ A convex hull can be represented by a set of linear inequalities

- Treat the set of all possible differential patterns of an S-box as a set of points in R^n. For example, the PRESENT S-box:

{(0, 0, 0, 0, 0, 0, 0, 0), (0, 0, 0, 1, 0, 0, 1, 1), (0, 0, 0, 1, 0, 1, 1, 1),
(0, 0, 0, 1, 1, 0, 0, 1), (0, 0, 0, 1, 1, 1, 0, 1), (0, 0, 1, 0, 0, 0, 1, 1),
(0, 0, 1, 0, 0, 1, 0, 1), (0, 0, 1, 0, 0, 1, 1, 0), (0, 0, 1, 0, 1, 0, 1, 0),
(0, 0, 1, 0, 1, 1, 0, 0), (0, 0, 1, 0, 1, 1, 0, 1), … }

Corresponds to the differential: 0010 → 1101

- Then we can compute the linear inequalities representation (H-representation) of the set of differential patterns

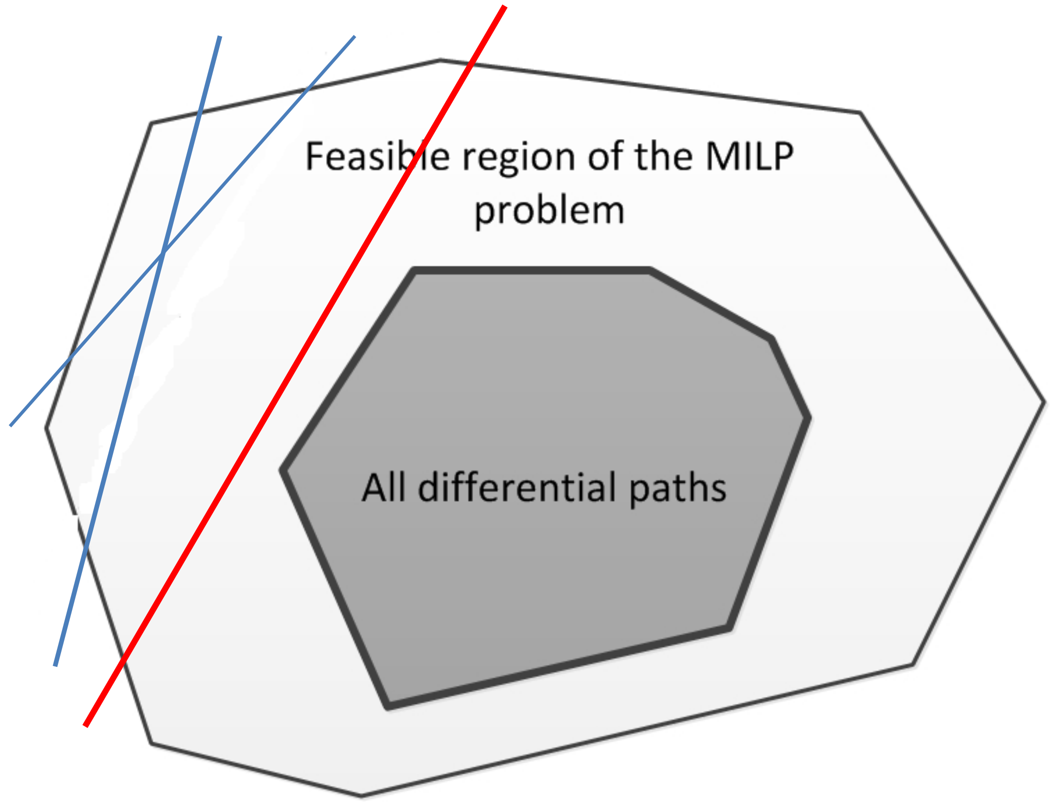
Linear inequality description of the PRESENT S-box.

Too many inequalities, which will make the MILP problem too difficult to be solved in practical time

```
( 0,  2, -2,  1, -1, -1, -2, -2,  6)( 0, -2,  2,  1, -2, -1, -1, -2,  6)( 0,  1, -1,  1,  0, -1, -1, -1,  3)(-1,  0,  1, -1, -1, -1,  0, -1,  4)
( 1,  0,  1,  1,  1,  1, -1,  0,  0)( 0,  1,  1,  1, -1,  0,  0,  1,  0)(-1, -1,  0,  1, -1,  0, -1, -1,  4)( 1,  0, -1,  2,  1,  2,  2, -1,  0)
( 1, -1, -1,  0,  0, -1,  1, -1,  3)(-1,  0, -1, -1,  1,  0, -1,  0,  3)( 0,  1,  1,  0, -1,  1, -1,  1,  1)( 1,  1,  1,  0,  0,  0, -1,  1,  0)
( 2,  1,  2,  2,  0,  1, -1, -1,  0)(-1,  0,  0,  1,  1,  1,  1,  0,  0)(-1,  0,  0,  1,  1,  0,  1,  1,  0)( 0,  1,  1, -1, -1,  1, -1,  0,  2)
( 0,  0, -1,  0,  0,  0,  0,  0,  1)( 1,  2,  0, -1, -2, -2, -2, -1,  6)( 0,  1, -1, -1, -1, -1,  0,  1,  3)( 0,  0, -1,  0,  1,  1,  1,  1,  0)
( 0,  1,  1, -2, -2, -1, -2, -2,  7)( 2,  2, -1,  2, -1,  3,  2, -1,  0)( 1,  0,  1, -1,  0,  1,  1,  1,  0)( 0,  1, -1,  2,  1,  2,  2, -1,  0)
(-2,  1,  1,  3,  1, -1,  1,  2,  0)( 1, -1, -1,  1,  1,  0,  0, -1,  2)( 0,  1,  1, -1,  1,  1,  1,  0,  0)(-2,  1,  1,  2,  1,  0,  1,  1,  0)
( 0, -1, -1, -1, -1,  1,  0, -1,  4)(-1,  1, -2, -1, -2, -2,  1,  2,  6)( 2, -2,  3, -4, -1, -4, -4,  1, 11)( 1, -1,  1,  2,  2,  2,  0, -1,  0)
( 0, -1,  1, -1,  0, -1, -1,  1,  3)( 0,  1, -1,  1,  0,  1,  1, -1,  1)( 1, -1,  2, -1,  0,  2,  2,  2,  0)(-1,  0, -1,  1, -1,  0, -1, -1,  4)
( 0, -1, -1,  0, -1,  1, -1,  1,  3)( 2,  1,  1,  0, -2, -1, -2, -1,  4)( 1,  0, -1,  1,  0,  1,  1, -1,  1)( 1,  2,  2,  1,  0,  0, -2,  1,  0)
( 0,  0,  0,  0,  0, -1,  0,  0,  1)( 1,  1,  1,  1,  1,  1,  1, -2,  0)( 1,  0,  1, -1,  1,  1,  0,  1,  0)( 0, -1,  1, -1, -1, -1, -1,  0,  4)
( 2,  1,  1, -2,  1,  1,  0,  2,  0)(-2,  1, -1, -1,  1, -1, -1, -1,  5)( 1, -1,  0,  2,  2,  2,  1, -1,  0)( 3,  2,  2, -1, -4, -2, -4, -1,  8)
( 0,  0,  0,  1,  1, -1,  1,  1,  0)( 0, -1, -1, -1,  1,  0, -1, -1,  4)( 2,  2,  1,  2, -1,  1,  0, -1,  0)( 1, -1, -1,  0,  1, -1,  0, -1,  3)
( 1, -1, -1,  0, -1,  0,  1, -1,  3)(-2,  0,  0,  1,  2,  1,  2,  1,  0)( 1, -1,  1,  2, -2, -2,  1, -2,  5)( 1, -1,  0,  0,  1,  1, -1, -1,  2)
( 3,  2,  3,  3, -1, -1,  0, -1,  0)( 1,  1, -1,  1, -1,  2,  1, -1,  1)( 1, -1, -1,  1,  2,  0,  2,  1,  0)( 1, -1, -1,  1,  0,  0,  1, -1,  2)
( 1,  1,  1,  1,  1,  1, -2,  1,  0)( 2,  1,  2, -1,  1,  0, -1,  2,  0)( 0, -1, -1, -1, -1,  0,  1, -1,  4)(-2,  1, -2, -1,  1, -1, -2,  0,  6)
( 0, -1,  2, -1, -2, -2, -2, -1,  7)(-1,  0, -1, -1,  1, -1,  0, -1,  4)( 3, -3, -2,  1, -1, -1,  3, -3,  7)( 1,  1, -1,  1,  0,  1,  1,  0,  0)
( 1,  1, -1, -1, -1, -1,  0,  0,  3)( 1, -2, -2,  0,  1, -2,  1, -1,  5)(-1, -1, -1, -1,  3,  4,  3,  4,  0)( 2,  2,  2,  0, -1,  0, -1,  1,  0)
( 2,  0,  2, -1,  2,  1, -1,  2,  0)( 1, -1, -2, -1, -2,  0,  2, -2,  6)( 1, -1,  0, -1,  1,  0, -1, -1,  3)( 1,  0, -1, -1,  0, -1,  1, -1,  3)
( 1,  1,  1,  1,  0,  0, -1,  0,  0)( 1,  0,  1,  1,  1,  1,  0, -1,  0)( 0,  0,  0, -1,  0,  0,  0,  0,  1)( 1, -2, -2, -1,  2, -3,  1, -2,  7)
( 1,  0,  1,  0, -1,  1,  1,  1,  0)(-1,  1,  1, -2, -1, -1, -1, -2,  6)( 1,  2, -1, -2, -2, -1,  1,  1,  4)( 2,  2, -1,  1, -1,  2,  2,  0,  0)
( 1,  1,  0,  1, -1,  1,  0, -1,  1)( 0,  0,  0, -1,  1,  1,  1,  1,  0)( 1,  1, -1,  0,  0,  1,  1,  1,  0)( 1, -1,  2, -2,  1, -1, -2,  1,  4)
( 1,  1, -1,  0, -1,  1,  1,  0,  1)(-3,  2, -2, -1,  1, -2, -2, -1,  8)( 0,  0,  1, -1, -1, -1, -1, -1,  4)( 0,  0,  0,  1,  0,  0,  0,  0,  0)
(-1,  0, -1, -1,  3,  3,  2,  3,  0)( 0,  1, -1, -1, -1, -1, -1,  0,  4)(-1,  0, -1, -1, -1, -1,  1,  1,  4)(-1,  1, -1,  0, -1, -1,  1,  1,  3)
( 1, -1,  1,  0,  0,  1,  1,  1,  0)( 1, -1,  1,  0,  1,  1, -1,  0,  1)(-4,  1,  1,  2,  3,  2,  3,  1,  0)( 2,  1,  1, -3,  1,  2,  1,  2,  0)
( 0,  0, -1,  1,  1,  1,  1,  0,  0)(-1,  0, -1, -1,  1,  1,  0,  1,  2)( 0,  0, -1,  1,  1,  0,  1,  1,  0)( 0,  1, -1,  1, -1,  0, -1, -1,  3)
( 1, -1,  0, -1,  1, -1,  0, -1,  3)(-1,  1,  0, -1,  0, -1, -1, -1,  4)( 0, -1,  0,  0,  0,  0,  0,  0,  1)( 0, -1, -1, -1, -1,  1, -1,  0,  4)
( 1,  1, -1,  2,  1, -2, -2, -2,  5)( 1,  1, -1, -1, -1,  0,  1,  0,  2)(-1,  1,  0, -1,  1, -1,  0, -1,  3)( 0, -1,  0,  0,  1,  1,  1,  1,  0)
( 0,  1,  1, -1, -1,  0, -1, -1,  3)( 0,  1, -1,  0, -1, -1, -1, -1,  4)( 2, -3, -1, -1,  2,  1, -3, -3,  8)( 0,  0,  0,  0,  0,  0,  0, -1,  1)
( 1,  2, -1, -2, -2, -2, -1,  0,  6)(-1,  1,  1,  2,  0, -1,  1,  2,  0)( 1,  1, -2,  1,  1,  1,  1,  1,  0)( 1, -1,  1,  1,  1,  1,  0,  0,  0)
( 0, -1, -1, -1,  1, -2,  1, -2,  5)( 1,  3, -2, -2,  3,  4,  1,  4,  0)(-1, -1, -1, -1,  0,  1,  0,  1,  3)(-2,  0,  2, -2, -1, -1,  1, -2,  6)
( 1, -2,  1,  0,  1,  2,  1,  2,  0)( 1, -1, -2, -2,  1, -3,  2, -2,  7)( 0, -2, -2,  3,  4,  1,  4,  1,  0)( 0,  0,  0, -1,  1, -1,  1, -1,  2)
(-1,  1,  1,  1,  0,  0,  0,  1,  0)(-1,  0,  1,  1,  0, -1,  1,  1,  1)( 0,  2,  2,  1, -1,  1, -1,  2,  0)( 1,  2, -1, -2, -2, -2,  0,  1,  5)
(-1,  1,  1, -1,  1, -1,  1, -2,  3)( 1,  1,  0,  1, -1,  1,  1,  0,  0)( 2,  2,  0, -1, -1,  1,  2,  2,  0)( 2,  1, -2, -2, -1, -1,  1, -1,  5)
( 4,  1,  3, -2,  3,  1, -2,  4,  0)( 1,  0,  0, -1, -1, -1, -1,  1,  3)( 0,  0, -2,  1,  2,  1,  2,  1,  0)( 0,  0,  0,  0, -1,  0,  0,  0,  1)
( 2,  2,  2,  1,  1, -3,  1,  1,  0)( 1,  1,  0, -2,  1,  2,  1,  2,  0)( 0, -1,  1,  1,  1,  1,  0, -1,  1)(-1,  1,  0,  1,  0, -1,  1,  1,  1)
( 1, -1,  1, -1,  1,  0, -1,  0,  2)( 1, -2, -1, -2,  2, -3,  1, -2,  7)( 1,  1,  0, -1, -1,  0,  1,  1,  1)( 1,  1,  1, -2,  1,  1,  1,  1,  0)
(-1, -1,  1,  0,  1, -1, -1,  1,  3)( 1,  1,  1, -1,  1,  0,  1,  0,  0)( 1,  1,  0, -1,  0,  1,  1,  1,  0)(-2, -1, -1,  2, -2,  0, -2, -1,  7)
( 1,  1,  0, -1, -1, -1, -1,  0,  3)( 1, -1, -1,  0,  1,  0,  1,  1,  1)( 0, -1,  1,  1,  0,  1, -1, -1,  2)( 3,  1,  1, -2, -2, -2, -2,  1,  5)
(-2, -1,  2, -1, -2, -2,  0, -1,  7)( 1,  1,  1,  0, -1,  0, -1,  0,  1)( 2,  0,  2,  1, -2, -1, -1, -2,  4)( 1,  0,  1,  0, -1, -1, -1, -1,  3)
(-2,  2,  1,  4,  1, -2,  2,  3,  0)(-1, -1,  1,  0, -1, -1,  0, -1,  4)( 1,  1,  0,  1,  0, -1, -1, -1,  2)( 1, -1, -1,  0,  1, -1,  1,  0,  2)
( 0, -1, -1,  1,  1,  0,  1,  0,  1)(-1, -1, -1, -1, -1,  1,  0,  0,  4)( 1,  0, -1, -1, -1,  0,  1, -1,  3)( 1, -2,  3, -2,  1,  4,  3,  4,  0)
( 1,  1,  0, -1,  1,  1,  0,  1,  0)( 1,  4, -1, -2, -4, -4, -3, -2, 12)( 0,  0,  0,  0,  0,  0,  0,  1,  0)( 0, -1,  1,  1, -1,  0, -1, -1,  3)
( 0,  1,  0,  0,  0,  0,  0,  0,  0)( 3, -1,  3, -1,  3,  2, -1,  2,  0)( 1,  1,  1,  1, -2,  1,  1,  1,  0)( 1,  0,  1,  1, -1, -1,  0, -1,  2)
( 0, -1, -1, -1,  1, -1,  0, -1,  4)( 2, -2,  1, -2,  1, -1, -1, -1,  5)( 1, -1,  3, -1, -2, -3, -3, -2,  9)( 1,  0,  1, -1, -1, -1, -1,  0,  3)
( 0, -1, -1,  1, -1,  0, -1,  1,  3)(-1,  0,  1,  2,  1, -1,  1,  2,  0)( 1,  0,  1, -1,  0, -1, -1,  1,  2)( 1,  0,  2, -1, -2, -2, -2, -1,  6)
( 1,  0,  1, -2,  1,  2,  1,  2,  0)( 0, -1,  0,  1,  1,  1,  1,  0,  0)( 1, -1, -1,  0,  1,  0, -1, -1,  3)( 0, -1,  0,  1,  1,  0,  1,  1,  0)
(-1,  1, -1, -1,  0, -1, -1,  0,  4)(-1,  0,  1, -1,  0, -1,  1, -1,  3)( 2,  1,  1, -1, -2, -1, -2,  0,  4)( 2, -1,  2,  1,  2,  2, -1,  0,  0)
( 1,  2,  2,  1,  0, -2,  0,  1,  0)( 1,  0,  0,  0,  0,  0,  0,  0,  0)( 1,  0,  1,  1,  0,  1, -1, -1,  1)( 2, -1, -3, -1, -3,  1,  2, -3,  8)
( 1, -1,  2, -2,  0, -2, -2,  1,  5)(-1, -1,  0, -1,  2,  3,  3,  3,  0)( 0,  1,  1,  1,  0, -1,  0,  1,  0)( 1,  1,  1,  0, -1,  0,  0,  1,  0)
( 1,  0,  1, -1,  1,  0, -1,  1,  1)(-2,  1,  2,  4,  1, -2,  2,  3,  0)( 1, -1,  4, -2, -3, -4, -4, -2, 12)( 1,  1,  0,  0,  1,  1, -1,  1,  0)
(-1, -1, -1,  0, -1,  1, -1,  0,  4)(-1, -3,  2,  1, -3, -2, -1, -3, 10)( 3,  3,  2,  3,  0, -1, -1, -1,  0)( 1,  1,  0,  0, -1,  1,  1,  1,  0)
( 0,  0,  0,  0,  0,  1,  0,  0,  0)( 0, -1, -1,  2,  2,  1,  2,  0,  0)( 4,  3,  1, -2, -2,  1,  3,  4,  0)( 0,  2, -1, -1, -2, -2, -2, -1,  7)
( 0,  1,  1,  2,  1, -2,  1,  1,  0)( 1,  1,  1,  1,  0, -1,  0,  0,  0)( 0, -1, -1,  2,  2,  0,  2,  1,  0)(-1, -1,  0, -1, -1,  0,  1,  0,  3)
( 1,  1, -1,  2,  0,  2,  2, -1,  0)( 1,  3, -1, -1, -3, -3, -2, -2,  9)( 1, -1, -1,  2,  1,  1,  1, -1,  1)( 0, -1, -1, -1,  0, -1,  1, -1,  4)
( 2,  2,  1, -1, -1,  0,  1,  2,  0)( 1, -2,  2, -3, -1, -3, -3,  1,  9)( 3,  4,  4,  1, -2,  0, -2,  1,  0)( 0,  1,  1,  0,  1,  0,  1, -1,  0)
( 2,  1,  1,  0, -2,  1,  1,  2,  0)( 0, -2,  0,  1,  2,  1,  2,  1,  0)( 1, -2, -1, -1,  2, -2,  0, -2,  6)(-2,  2,  0, -2,  1, -1, -1, -2,  6)
( 0, -1, -1, -1,  0,  1, -1, -1,  4)(-2, -1, -1,  1, -2,  1, -2, -1,  7)(-1,  2, -3,  1, -1, -2, -3, -3, 10)( 1,  2, -1, -1,  2,  2,  0,  2,  0)
(-3, -2,  2, -1, -2, -2,  1, -1,  8)( 0, -1,  1,  0, -1, -1, -1, -1,  4)( 1, -1, -1,  3,  2,  2,  2, -1,  0)( 1, -1,  0,  1,  1,  1,  0, -1,  1)
( 2,  3, -2, -4, -4, -4, -1,  1, 11)( 2,  0,  1, -2, -1, -2, -2,  1,  5)(-1,  1, -1,  0,  1, -1, -1,  0,  3)( 2,  3,  3,  2,  1, -4,  1,  1,  0)
(-1,  1,  0, -1,  1,  0, -1, -1,  3)(-1,  1,  1,  3,  1, -2,  1,  2,  0)( 0,  0,  0,  0,  0,  0, -1,  0,  1)(-2,  2, -1, -1,  0, -2, -2, -1,  7)
( 1,  1,  1,  0,  0, -1,  0,  1,  0)( 1,  1,  0,  0, -1, -1, -1, -1,  3)( 1,  2, -2, -3, -3, -3, -1,  1,  9)( 1, -2,  1,  1,  1,  1,  1,  1,  0)
(-3,  1,  1,  1,  2,  2,  2,  1,  0)( 1,  2,  2,  0, -1,  1, -1,  1,  0)( 1, -2, -3, -2,  1, -4,  3, -3, 10)(-1,  0,  1, -1, -1,  0,  1, -1,  3)
( 0, -1,  1,  2,  2,  2,  1, -1,  0)( 0,  1,  1,  1,  0,  0, -1,  1,  0)( 3,  2,  2,  2, -1,  0, -1, -1,  0)(-1, -1, -1, -1,  0,  1, -1,  0,  4)
( 2,  2,  2, -1,  3, -1,  3, -1,  0)(-1,  1, -1,  0,  0, -1, -1, -1,  4)( 2,  1,  1,  1, -1,  0, -1, -1,  1)( 2,  2,  0,  1, -1, -1, -2, -2,  4)
( 0,  0,  0,  1,  1,  1,  1, -1,  0)(-1,  0,  0,  0,  0,  0,  0,  0,  1)( 1, -1,  1,  1,  1,  2, -1, -1,  1)( 0,  0,  0,  0,  0,  0,  1,  0,  0)
(-1, -2, -2, -1, -1,  2, -1,  0,  6)( 1,  0, -1,  0, -1,  1,  1, -1,  2)( 0, -1,  1, -1,  0,  1,  1,  1,  1)( 0, -1,  1, -1,  1,  2,  2,  2,  0)
( 4,  3,  3, -1, -1, -1, -1,  3,  0)(-1, -1, -1,  1, -1,  0, -1,  0,  4)( 1,  1, -1,  0,  1,  1,  0,  1,  0)( 1,  1,  1,  1,  0,  0,  0, -1,  0)
(-1, -1,  0, -1,  0,  1,  1,  1,  2)( 1,  1,  0, -1, -1, -1,  0,  1,  2)(-2,  1,  2,  4,  2, -2,  1,  3,  0)(-2, -2,  1, -1, -2, -1,  1,  0,  6)
( 0, -1,  1,  1, -1, -1,  0, -1,  3)( 1,  1,  1, -1,  0,  0,  0,  1,  0)( 1,  2,  2,  1, -2,  0,  0,  1,  0)(-1, -1,  1,  0, -1, -1,  1,  0,  3)
( 3,  3,  2, -1, -1, -1,  0,  3,  0)(-1,  0,  0,  0,  1,  1,  1,  1,  0)( 3,  3, -1, -1, -1,  2,  3,  2,  0)( 2, -1,  2,  0,  2,  2, -1,  1,  0)
(-1, -1,  0, -1,  0, -1,  1, -1,  4)(-2, -1,  1, -1, -1, -1,  1, -1,  5)( 1,  1, -2,  0,  1,  2,  1,  2,  0)( 1, -1, -1,  0, -1,  0, -1,  1,  3)
( 0,  1,  0, -1, -1, -1, -1, -1,  4)( 2,  1,  1,  0,  1,  1, -2,  2,  0)(-2,  2,  1,  4,  2, -2,  1,  3,  0)( 1, -2, -2, -1,  1, -3,  2, -2,  7)
( 1,  0,  1,  0,  1,  1, -1,  1,  0)( 1, -2, -1, -1,  2,  0, -2, -2,  6)(-1,  0,  1,  1,  1, -1,  0,  1,  1)( 2,  1,  0, -2, -2, -2, -1,  1,  5)
( 1, -1, -2, -1,  0, -2,  2, -2,  6)(-1,  1,  1, -1,  0,  0,  0, -1,  2)( 1,  1,  1,  1, -1,  0,  0,  0,  0)( 1, -1, -1, -1,  1, -2,  1, -1,  4)
( 2,  1,  1, -2,  0,  1,  1,  2,  0)(-1, -1,  1, -1, -1, -1,  0,  0,  4)( 3,  2,  3, -1,  0, -1, -1,  3,  0)( 1, -1,  1,  0,  1,  1,  0,  1,  0)
( 1, -2, -2,  2,  1,  0,  1, -1,  3)( 1, -1,  0,  1, -1, -1,  1, -1,  3)(-1,  3,  3, -1,  2,  2,  2, -1,  0)(-1,  1,  0,  1,  1, -1,  0,  1,  1)
(-1, -2,  1, -1,  1, -2, -2,  2,  6)( 0,  1, -1,  1, -1,  1,  0, -1,  2)( 0,  1, -1, -1,  2,  2,  1,  2,  0)(-1, -1,  0, -1,  1, -1, -1,  1,  4)
( 0,  0,  0,  0,  1,  0,  0,  0,  0)( 1, -3, -2, -2,  3, -4,  1, -3, 10)( 2,  2, -1,  0, -1,  2,  2,  1,  0)( 1, -1,  2, -2, -1, -2, -2,  0,  6)
( 2, -1,  2,  2,  2,  3, -1, -1,  0)( 0, -2, -2, -2, -1,  2, -1, -1,  7)( 3, -2, -3,  1,  3, -1, -1, -3,  7)(-1,  1,  0,  2,  1, -1,  1,  2,  0)
( 0,  1, -1, -1,  1,  1,  0,  1,  1)( 1,  1,  0,  1,  0,  1,  1, -1,  0)( 1, -1,  1, -1,  0, -1, -1,  0,  3)( 1, -1, -1,  0,  2,  1,  2,  2,  0)
( 0,  0,  1,  0,  0,  0,  0,  0,  0)( 1,  0, -1,  1,  1, -1, -1, -1,  3)(-1,  1,  1,  2,  1, -1,  0,  2,  0)
```

Our method: mixed-integer programming based

- ☐ Convex hull computation
  - ✓Can we use less inequalities？ Yes！



Our method: mixed-integer programming based

☐ Convex hull computation

✓Can we use less inequalities ?   Yes!

Algorithm 1: Selecting n inequalities from the convex hull $\mathcal{H}$ of an S-box

---

Input:
$\mathcal{H}$: the set of all inequalities in the H-representation of the convex hull of an S-box;
$\mathcal{X}$: the set of all possible differential patterns of an S-box;
n: a positive integer.
Output: $\mathcal{O}$: a set of n inequalities selected from $\mathcal{H}$

```
l* := None;
X* := X;
H* := H;
O := ∅;
for i ∈ {0, ..., n − 1} do
    l* := The inequality in H* which maximizes the number of removed
          impossible differential patterns from X* ;
    X* := X* − {removed impossible differential patterns by l*};
    H* := H* − {l*};
    O := O ∪ {l*};
end
return O
```

Applications, Improvements and Adaptations for Specific Ciphers

☐ Applications

- ✓ Obtain security bounds w.r.t. (related-key) differential attack
- ✓ Search for or enumerate characteristics with specific properties

☐ Improvements and Adaptations

- ✓ Search for the best characteristic of some ciphers with small S-boxes, e.g. 4×4 S-boxes
- ✓ Construct MIP model whose feasible region is exactly the set of all valid differential characteristics for SIMON like ciphers
- ✓ …

Applications

- Obtain security bounds
  - ✓ Set the objective function to minimize the number of active S-boxes. The optimized solution will give a lower bound of the #Active S-boxes

- Search for or enumerate characteristics
  - ✓ Every feasible solution corresponds to a characteristic
  - ✓ After find a solution, add a constraint to the MIP model such that this solution is not feasible any more and find another solution. Repeating this again and again we can enumerate the feasible region

Improvements and Adaptations for Specific Ciphers

- Search for the best characteristic of some ciphers with small S-boxes, e.g. 4×4 S-boxes
  - ✓ The idea is to encode the probability information of into the differential pattern of the S-box

Search for the best characteristic of ciphers with 4×4 S-boxes

- For every differential pattern $(x_0,\cdots,x_3,y_0,\cdots,y_3)$ of a 4×4 S-box, a corresponding differential pattern with probability information can be constructed in the following way

$$(x_0,x_1,x_2,x_3,y_0,y_1,y_2,y_3;p_0,p_1)\in\{0,1\}^{8+2}$$

$$\begin{cases}(p_0,p_1)=(0,0),\ \text{if } \Pr_S[(x_0,\ldots,x_{\omega-1})\to(y_0,\ldots,y_{\nu-1})]=1;\\(p_0,p_1)=(0,1),\ \text{if } \Pr_S[(x_0,\ldots,x_{\omega-1})\to(y_0,\ldots,y_{\nu-1})]=2^{-2};\\(p_0,p_1)=(1,1),\ \text{if } \Pr_S[(x_0,\ldots,x_{\omega-1})\to(y_0,\ldots,y_{\nu-1})]=2^{-3}.\end{cases}$$

- That is, use the extra two bit (p_0, p_1) to encode the probability information, and the probability of the differential pattern

$$(x_0,\cdots,x_3,y_0,\cdots,y_3)$$

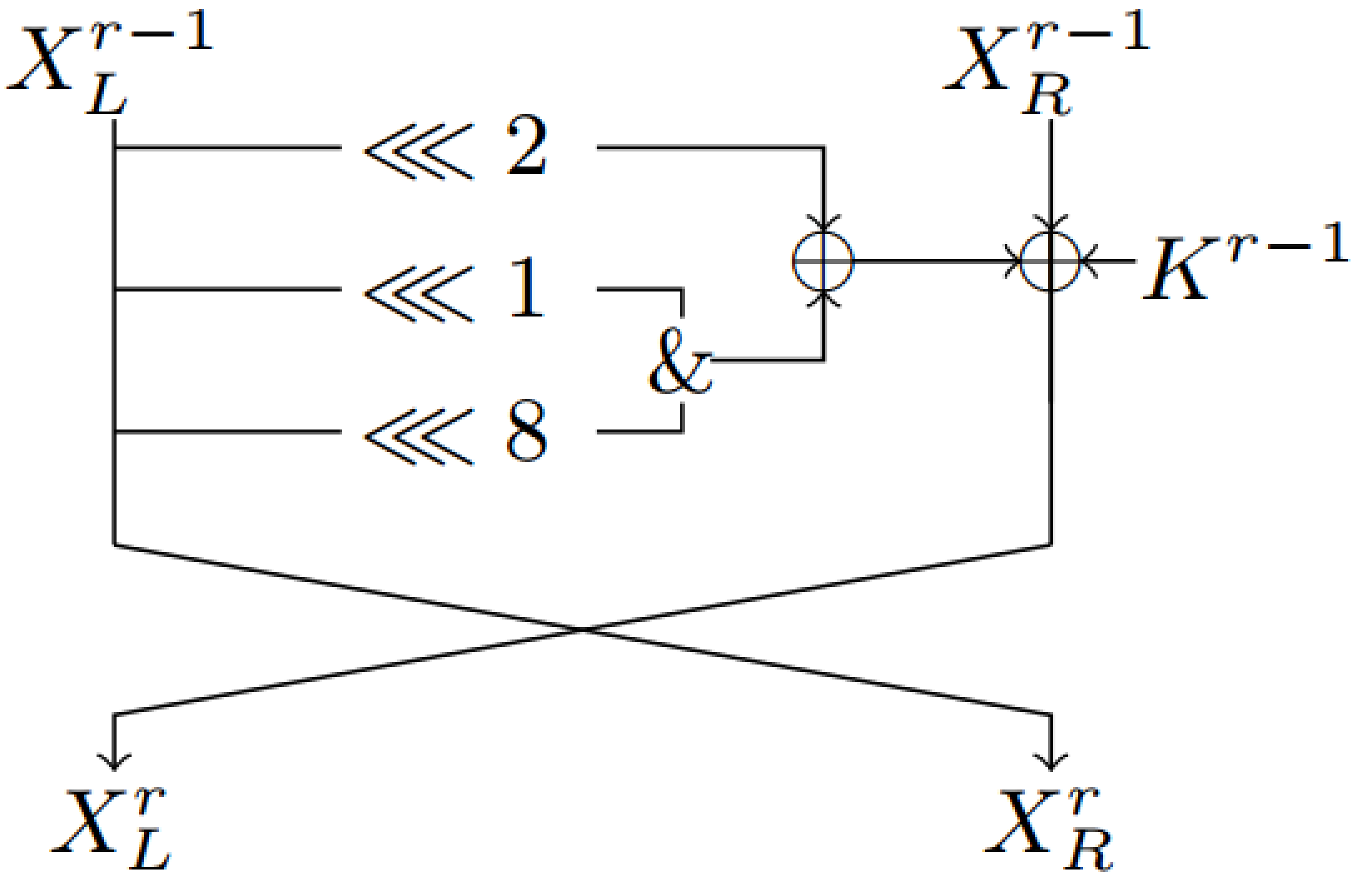
is $2^{-(p_0+2p_1)}$

- Set the objective function to be Minimize the sum of all p_0+2p_1

Improvements and Adaptations for Specific Ciphers

- Improvements and Adaptations
  - ✓ Construct MIP model whose feasible region is exactly the set of all valid differential characteristics for SIMON like ciphers
  - ✓ …

Exact Feasible Region for SIMON-like Ciphers



Exact Feasible Region for SIMON-like Ciphers

- The nonlinear layer of SIMON32 can be described by a nonlinear function $F : \mathbb{F}_2^{16} \rightarrow \mathbb{F}_2^{16}$

$$F(x) = (x <<< 1) \cdot (x <<< 8), \ x = (x_0, \cdots, x_{15}) \in \mathbb{F}_2^{16}$$

- Let $\Delta = (\Delta_0, \cdots, \Delta_{15}) \in \mathbb{F}_2^{16}$, and $\delta = (\delta_0, \cdots, \delta_{15}) \in \mathbb{F}_2^{16}$, then the differential $\Delta \rightarrow \delta$ is a valid if and only if the following system of equation has a solution

$$\left\{\begin{array}{l}
\delta_0 = \Delta_1 \cdot x_8 + \Delta_8 \cdot x_1 \\
\delta_1 = \Delta_2 \cdot x_9 + \Delta_9 \cdot x_2 \\
\delta_2 = \Delta_3 \cdot x_{10} + \Delta_{10} \cdot x_3 \\
\delta_3 = \Delta_4 \cdot x_{11} + \Delta_{11} \cdot x_4 \\
\delta_4 = \Delta_5 \cdot x_{12} + \Delta_{12} \cdot x_5 \\
\delta_5 = \Delta_6 \cdot x_{13} + \Delta_{13} \cdot x_6 \\
\delta_6 = \Delta_7 \cdot x_{14} + \Delta_{14} \cdot x_7 \\
\delta_7 = \Delta_8 \cdot x_{15} + \Delta_{15} \cdot x_8
\end{array}\right.
\qquad
\left\{\begin{array}{l}
\delta_8 = \Delta_9 \cdot x_0 + \Delta_0 \cdot x_9 \\
\delta_9 = \Delta_{10} \cdot x_1 + \Delta_1 \cdot x_{10} \\
\delta_{10} = \Delta_{11} \cdot x_2 + \Delta_2 \cdot x_{11} \\
\delta_{11} = \Delta_{12} \cdot x_3 + \Delta_3 \cdot x_{12} \\
\delta_{12} = \Delta_{13} \cdot x_4 + \Delta_4 \cdot x_{13} \\
\delta_{13} = \Delta_{14} \cdot x_5 + \Delta_5 \cdot x_{14} \\
\delta_{14} = \Delta_{15} \cdot x_6 + \Delta_6 \cdot x_{15} \\
\delta_{15} = \Delta_0 \cdot x_7 + \Delta_7 \cdot x_0
\end{array}\right.$$

Exact Feasible Region for SIMON-like Ciphers

- Let $Sol(\delta_0 = \Delta_1 \cdot x_8 + \Delta_8 \cdot x_1)$ be the set of all 0-1 solutions for this equation.

- The vectors $(\delta_0, \Delta_1, x_8, \Delta_8, x_1)$ in $Sol(\delta_0 = \Delta_1 \cdot x_8 + \Delta_8 \cdot x_1)$ are given below

```
(0, 0, 0, 0, 0) (0, 0, 0, 0, 1)
(0, 0, 0, 1, 0) (0, 0, 1, 0, 0)
(0, 0, 1, 0, 1) (0, 0, 1, 1, 0)
(0, 1, 0, 0, 0) (0, 1, 0, 0, 1)
(0, 1, 0, 1, 0) (0, 1, 1, 1, 1)
(1, 0, 0, 1, 1) (1, 0, 1, 1, 1)
(1, 1, 0, 1, 1) (1, 1, 1, 0, 0)
(1, 1, 1, 0, 1) (1, 1, 1, 1, 0)
```

Exact Feasible Region for SIMON-like Ciphers

- The H-representation of the convex hull of $Sol(\delta_0 = \Delta_1 \cdot x_8 + \Delta_8 \cdot x_1)$ is given below

```
( 0, -1,  0,  0,  0, 1 ) ( 0,  0, -1,  0,  0, 1 )
( 0,  0,  0, -1,  0, 1 ) (-1,  1,  0,  0,  1, 0 )
(-1,  0,  1,  0,  1, 0 ) (-1,  0,  0,  0,  0, 1 )
( 0,  0,  0,  0,  1, 0 ) ( 1, -1, -1,  0,  1, 1 )
( 0,  1,  0,  0,  0, 0 ) ( 0,  0,  0,  0, -1, 1 )
(-1,  1,  0,  1,  0, 0 ) ( 1,  0,  0,  0,  0, 0 )
(-1,  0,  1,  1,  0, 0 ) ( 0,  0,  0,  1,  0, 0 )
( 1,  0,  1, -1, -1, 1 ) ( 0,  0,  1,  0,  0, 0 )
( 1, -1, -1,  1,  0, 1 ) ( 1,  1,  0, -1, -1, 1 )
(-1, -1, -1, -1, -1, 4 )
```

where a 6-dimensional vector $(\lambda_0, \cdots, \lambda_4, \gamma)$ denotes the linear inequality

$$\lambda_0\delta_0 + \lambda_1\Delta_1 + \lambda_2 x_8 + \lambda_3\Delta_8 + \lambda_4 x_1 + \gamma \geq 0$$

Exact Feasible Region for SIMON-like Ciphers

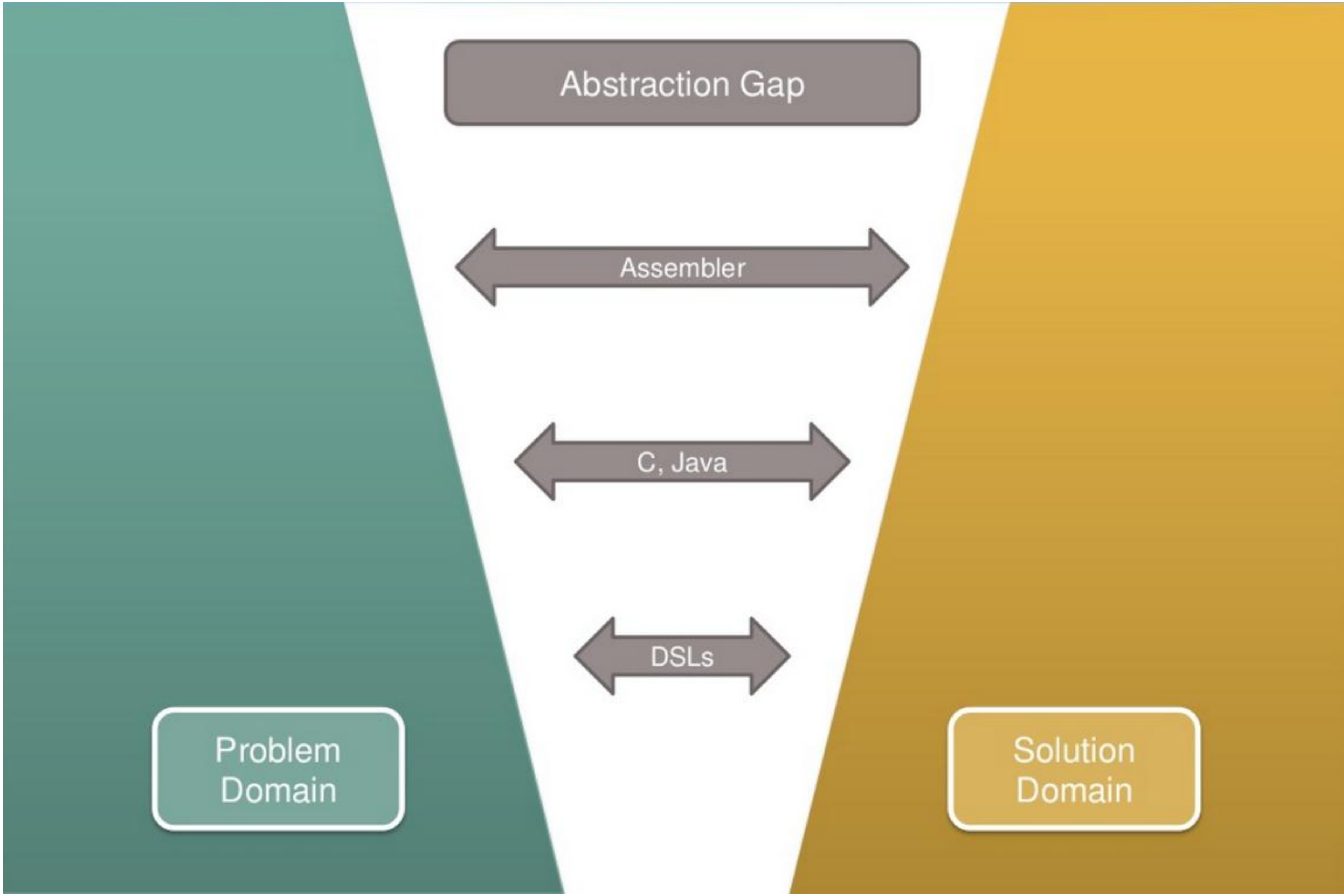
- From the H-representation we can derive the critical set (using the greedy algorithm)

```
(-1,  0,  1,  0,  1, 0)
(-1,  1,  0,  1,  0, 0)
( 1, -1, -1,  1,  0, 1)
( 1,  1,  0, -1, -1, 1)
( 1,  0,  1, -1, -1, 1)
(-1, -1, -1, -1, -1, 4)
(-1,  1,  0,  0,  1, 0)
(-1,  0,  1,  1,  0, 0)
( 1, -1, -1,  0,  1, 1)
```

Domain Specific Language (DSL) for Constraint Programming based Automatic Cryptanalysis

Domain Specific Language

- A Domain-Specific Language (DSL) is a computer language specialized to a particular application domain.
- Compared with General Purpose Language (GPL), DSL offers …
  - ✓ Higher abstractions
  - ✓ Avoid redundancy
  - ✓ Separation of concerns
  - ✓ Use domain concepts



- http://modeling-languages.com/introduction-to-domain-specific-languages-slides/

Domain Specific Language

Examples

- SQL

```sql
CREATE TABLE Employee (
  id INT NOT NULL IDENTITY (1,1) PRIMARY KEY,
  name VARCHAR(50),
  surname VARCHAR(50),
  address VARCHAR(255),
  city VARCHAR(60),
  telephone VARCHAR(15),
)
```

- CSS

```css
body {
  text-align: left;
  font-family: helvetica, sans-serif;
}
h1 {
  border: 1px solid #b4b9bf;
  font-size: 35px;}
```

- HTML

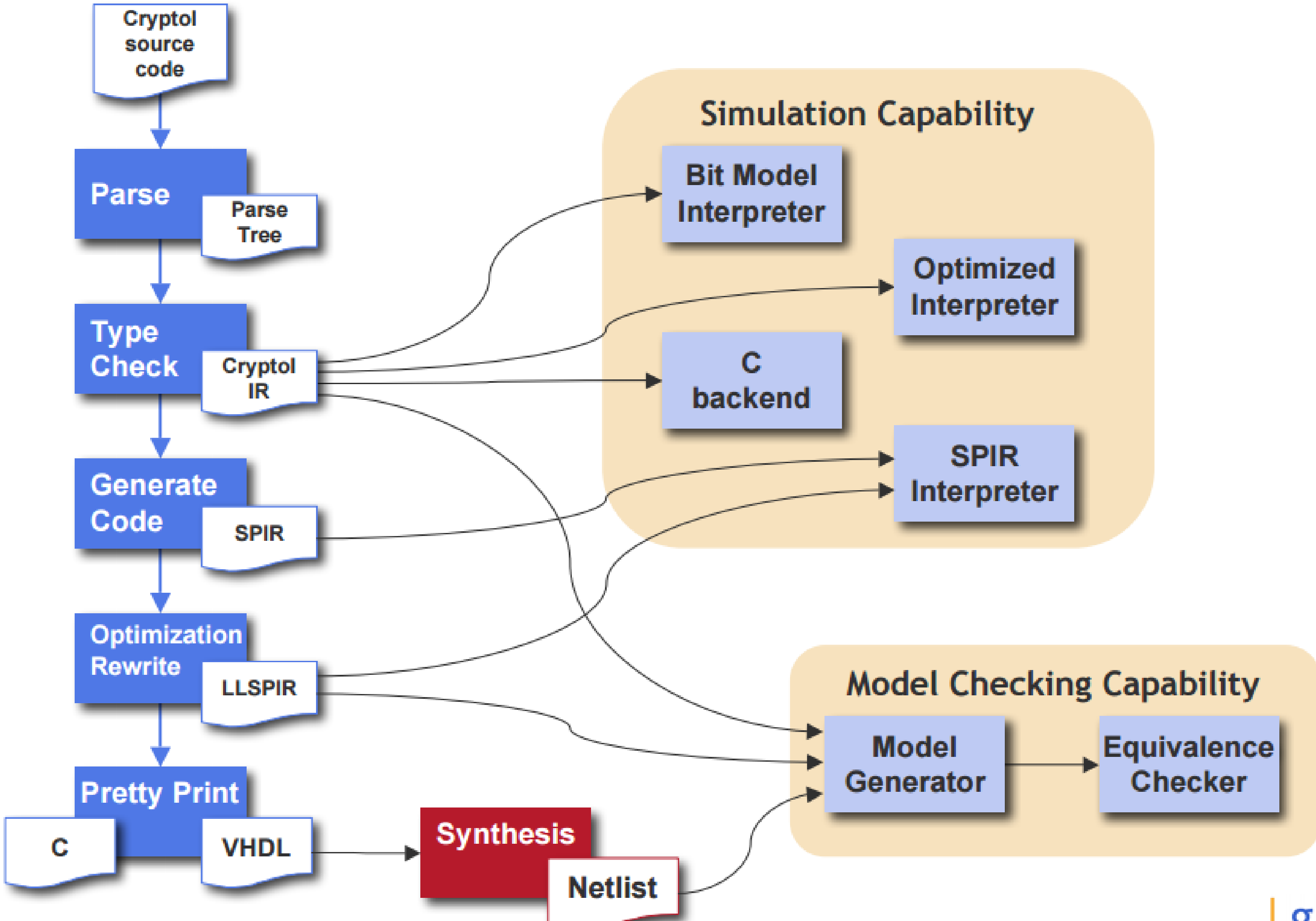
```html
<html>
  <head>
    <title>Example</title>
  </head>
  <body>
    <p>Example</p>
  </body>
</html>
```

- LaTeX

```latex
\ifthenelse{\boolean{showcomments}}
  {\newcommand{\nb}[2]{
    \fcolorbox{gray}{yellow}{
      \bfseries\sffamily\scriptsize#1
    }
    {\sf\small\textit{#2}}
    }
    \newcommand{\version}{\scriptsize$-$working$-$}
  }
  {\newcommand{\nb}[2]{}
    \newcommand{\version}{}
}
```

A Domain Specific Language for Crypto Implementation

- Cryptol



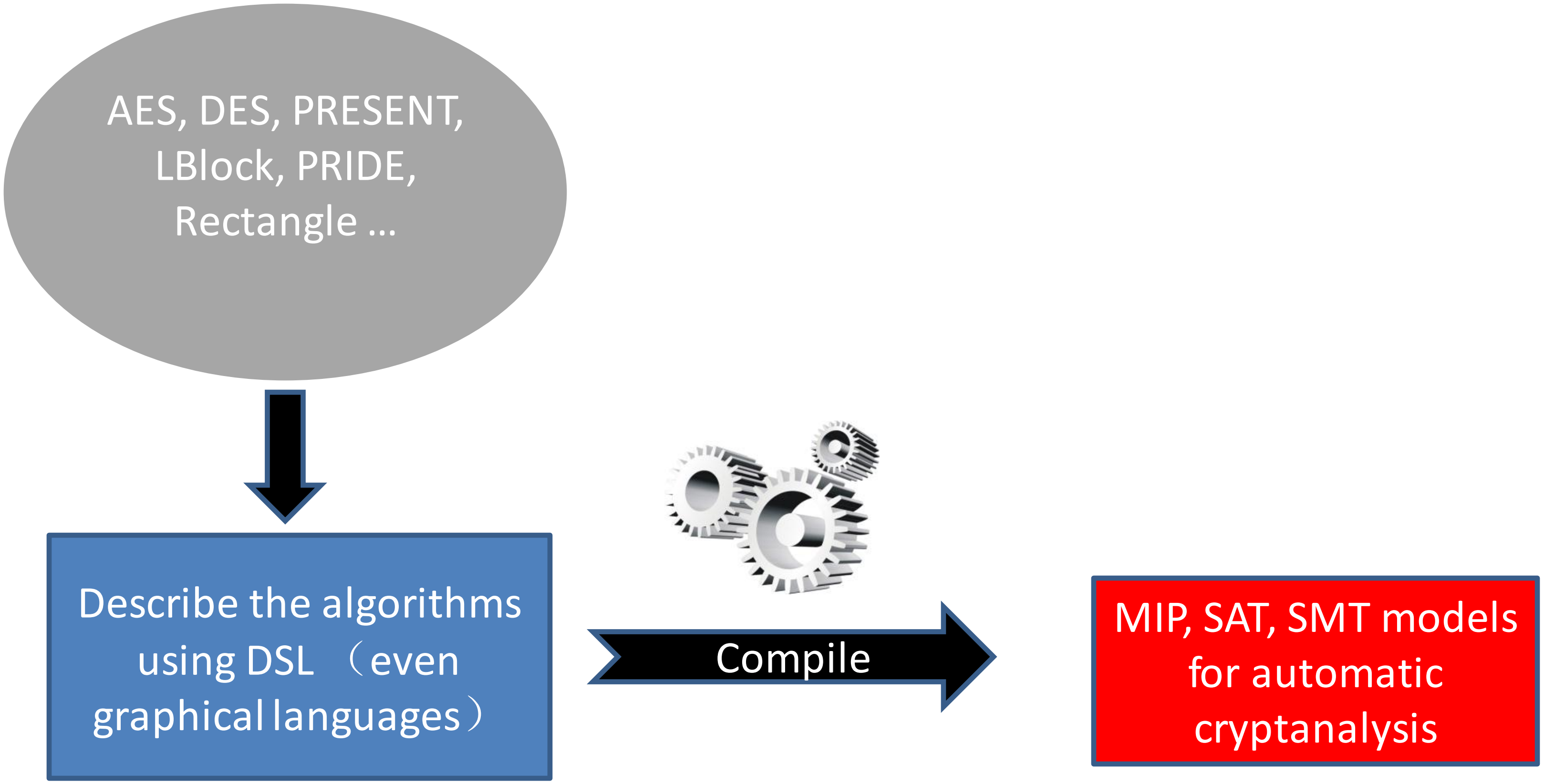
galois

Constraint Programming

- A mathematical problem involving a set of variables, constraints, and some objective.

- Constraint programming examples:
  - ✓ SAT problem
  - ✓ Mixed Integer Programming
  - ✓ Satisfiability Modulus Theory (SMT) problems
  - ✓ Constraint Integer Programming
  - ✓ …

- Already used in cryptographic research


AES, DES, PRESENT,
LBlock, PRIDE,
Rectangle …
Describe the algorithms
using DSL （even
graphical languages）
Compile
MIP, SAT, SMT models
for automatic
cryptanalysis

Main References

1. Mouha, Nicky, et al. "Differential and linear cryptanalysis using mixed-integer linear programming." Information Security and Cryptology. Springer Berlin Heidelberg, 2012.
2. Sareh Emami, San Ling, Ivica Nikolic, Josef Pieprzyk and Huaxiong Wang. The Resistance of PRESENT-80 Against Related-Key Differential Attacks. Cryptology ePrint Archive, Report 2013/522, 2013.
3. Wu, Shengbao, and Mingsheng Wang. "Automatic Search of Truncated Impossible Differentials for Word-Oriented Block Ciphers." Progress in Cryptology-INDOCRYPT 2012. Springer Berlin Heidelberg, 2012. 283-302.
4. Bouillaguet, Charles, Patrick Derbez, and Pierre-Alain Fouque. "Automatic search of attacks on round-reduced AES and applications." Advances in Cryptology–CRYPTO 2011. Springer Berlin Heidelberg, 2011. 169-187.
5. Biryukov, Alex, and Ivica Nikolić. "Automatic search for related-key differential characteristics in byte-oriented block ciphers: Application to AES, camellia, khazad and others." Advances in Cryptology–EUROCRYPT 2010. Springer Berlin Heidelberg, 2010. 322-344.
6. Wu, Shengbao, and Mingsheng Wang. Security evaluation against differential cryptanalysis for block cipher structures. Cryptology ePrint Archive, Report 2011/551, 2011.
7. Biryukov, Alex, and Ivica Nikolić: Search for related-key differential characteristics in DES-like ciphers. In: Fast Software Encryption – FSE 2011. pp. 18–34. Springer (2011)
8. Knudsen, Lars R., and Matthew Robshaw. The block cipher companion. Springer, 2011.
9. Mitsuru Matsui, On correlation between the order of S-boxes and the strength of DES, Eurocrypt 1994.
10. Siwei Sun, Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi, Ling Song: Towards

11. Finding the Best Characteristics of Some Bit-oriented Block Ciphers and Automatic Enumeration of (Related-key) Differential and Linear Characteristics with Predefined Properties. IACR Cryptology ePrint Archive 2014: 747 (2014)

Thanks！